ONDERZOEKSRAPPORT NR 8906

THE SCHEDULING OF ACTIVITIES TO MAXIMIZE THE NET PRESENT VALUE OF PROJECTS

BY

S.E. ELMAGHRABY
W.S. HERROELEN
E. GALLENS

D/1989/2376/9

THE SCHEDULING OF ACTIVITIES TO MAXIMIZE THE NET PRESENT VALUE OF PROJECTS

Salah E. ELMAGHRABY*
Willy S. HERROELEN**
Els GALLENS**

This research was partially supported by ARO Contract N° DAAL03-86-K-0039, which is gratefully acknowledged.

*Graduate Program in Operations Research, North Carolina State University, Raleigh, NC 27695-7913

**Department of Applied Economic Sciences, Katholieke Universiteit Leuven, Dekenstraat 2, B-3000 Leuven (Belgium)

THE SCHEDULING OF ACTIVITIES TO MAXIMIZE THE NET PRESENT VALUE OF PROJECTS

ABSTRACT

We review and critique the approaches offered for the resolution of the problem of maximizing the net present value of a project through the manipulation of the times of realization of its key events. We offer an optimal solution procedure that maintains the essential simplicity of the problem. The procedure has been programmed in C for personal computers running under DOS. Detailed computational results obtained on a problem set involving 250 randomly generated projects are reported to illustrate the efficiency of the procedure.

1. STATEMENT OF THE PROBLEM

The following problem has been posed, and treated, by several researchers in the field of activity networks (ANs): given specified "net cash flows" (ncf's) at selected nodes (the so-called "key events" (KE's)) of a project (in the activity-on-the-arc mode of representation), what is the optimal schedule of the realization times of these KE's (or, alternatively, the schedule of the activities leading to their realization) in order to maximize the net present value (NPV) of the project as a whole? Note that the ncf at node i, denoted by a_i, if different from zero, may be positive or negative, reflecting net receipts or net disbursements, respectively. In a typical project, positive and negative ncf's are interspersed, with the majority of the earlier cash flows being negative, reflecting outlays by the "contractor" which are not fully recovered by "owner" payments, and the majority of the later cash flows being positive, reflecting the recoupment by the contractor of his expenditures plus a reasonable profit.

The fundamental assumptions of all prior treatments are the following. First, it is assumed that the ncf a_i is independent of the time of realization of KE(i). This assumption is never explicitly stated, though it is evidently necessary for the validity of the subsequent derivations offered by the researchers. Second, it is assumed thet the ncf's $\{a_i\}$ are known a priori - eliminating any relevance of the analysis done to the "bidding problem" in ANs, since the latter is concerned with the very determination of the values of these ncf's.

We highlight these two basic assumptions at the outset because of the central role they play in our criticisms of the proposed approaches, to be presented in Section 3 after a brief summary of the literature given in the next section. In Section 4, we present an optimal solution procedure. Section 5 describes the computational results obtained by the procedure on some 250 randomly generated projects.

2. REVIEW OF LITERATURE

The vast majority of the so-called project scheduling methodologies presented in the literature have been developed with the objective of minimizing the project duration subject to precedence and resource constraints (the resource-constrained project scheduling problem). Alternatively, the problem is posed with the objective of leveling the resource usage for a given project duration (the resource leveling problem). An overview of both problems can be found in Elmaghraby (1977). In doing so, the financial aspects of project management are, unfortunately, largely ignored. When taken into consideration, there is a decided preference for the maximization of project NPV as the more appropriate objective, since the time value of money is taken into account. Evidently, the adoption of the NPV criterion presumes knowledge of the discount factor $\beta = 1/(1+r)$, where r is the hurdle rate or cost of capital; i.e., the return available for equivalent risk investments traded in the capital markets.

To the best of our knowledge, Russell (1970) was the first to introduce the idea of maximizing the NPV of the cash flows of a project under the stipulated assumptions mentioned above. Consider a project with m activities (m arcs in the activity-on-arrow mode of representation) with fixed durations $\{d_k\}$ $(k=1,\ldots,m)$, and n events (n nodes in the activity-on-arrow mode of representation), to occur at time instants $\{T_i\}$, with associated ncf's $\{a_i\}$ $(i=1,2,\ldots,n)$. His objective function is to

$$\text{maximize} \quad \Sigma_i \, a_i \exp(-\alpha T_i) \qquad [1]$$

where $e^{-\alpha} = 1/(1+r) = \beta$, the discount factor. For uniformity of expression, we sometimes re-write the criterion [1] as:

$$\text{maximize} \quad \Sigma_i \, a_i \, \beta^{T_i} \qquad [1']$$

This maximization is subject to the time precedence constraints

$$T_{i(k)} + d_k <= T_{j(k)} \quad , k=1,\ldots,m \qquad [2]$$

where i(k) and j(k) denote the tail and head nodes of activity k, respectively. Clearly, Eq. [1] (or [1']) represents the maximization of the NPV of the project, while Eqs. [2] represent the enforcement of the precedence relationships on the network activities.

The nonlinear programming problem of Eqs. [1]-[2] is solved by Russell iteratively through successive approximation in the following manner. Initially, the nonlinear objective function Eq. [1] is approximated by considering only the first, (linear) term of the associated Taylor series expansion. Assuming a current non-optimum but feasible solution given by the event times T°_i, we have, for T_i close to T°_i,

$$\Sigma_i a_i \exp(-\alpha T_i) \approx \Sigma_i a_i \exp(-\alpha T^{\circ}_i) - \Sigma_i (T_i - T^{\circ}_i) a_i \alpha \exp(-\alpha T^{\circ}_i)$$

$$= \Sigma_i a_i \exp(-\alpha T^{\circ}_i) + \Sigma_i T^{\circ}_i a_i \alpha \exp(-\alpha T^{\circ}_i) - \Sigma_i T_i a_i \alpha \exp(-\alpha T^{\circ}_i)$$

$$= \text{constant} - \Sigma_i T_i a_i \alpha \exp(-\alpha T^{\circ}_i)$$

and the original objective Eq. [1] is replaced by the maximization of the linear objective :

$-\Sigma_i T_i a_i \alpha \exp(-\alpha T^{\circ}_i) = -\Sigma_i T_i a_i \alpha \beta^{T_i}$, subject to the precedence constraints Eqs. [2].

The dual form of this (primal) linear programming (LP) model turns out to be a transshipment problem over a network flow model. The solution of this transshipment model yields a system of flows. By the complimentary slackness principle of LP, flows will only occur in arcs whose corresponding primal activity has no float. As such, the flows on the arcs of the transshipment model impute an occurrence time for each node of the network. These imputed event times are then utilized in the Taylor series expansion to provide an improved linear approximation to the primal nonlinear objective function. The parameters of the associated dual LP are subsequently updated and the solution of the updated (with new times) transshipment model yields a new set of event realization times. The whole process is repeated until successive event times at all nodes are identical. Russell

provides proof that the resulting node times do converge, and their point of accumulation constitutes at least a local optimum of the original NPV maximization problem. Apart from an example illustrating the application of the algorithm, he does not report on any computational experience.

Grinold (1972) shows that the nonlinear program with linear constraints and convex objective of Eqs. [1]-[2] can be transformed into an equivalent linear program (Grinold's treatment is equally valid when applied to non-concave objective functions). This fact is then used to demonstrate that the optimal solution of the scheduling problem corresponds to a feasible tree in the project diagram which consists of all the arcs having no float, i.e., an extreme point in the set of feasible schedules. As a consequence, Grinold restricts the search for optimal schedules to feasible trees in the project network. Using standard complementary slackness results for checking the optimality of feasible trees, he develops two solution procedures which are related to Markowitz' special procedure for the weighted distribution problem (Dantzig 1963), requiring the solution of triangular systems of equations with all matrix coefficients equal to ± 1 or 0. The first algorithm solves the problem for any given project deadline. The second algorithm finds the optimal solution for all possible project deadlines. This yields a curve explicitly showing the trade-off in project duration and present value. Again, apart from an example illustrating these computations, no further computational experience is given.

In addition to the just described procedures for maximizing the NPV of a project under the assumption that both positive and negative ncf's occur over the course of a project, a number of procedures have been developed for solving the problem under (additional) resource constraints. They are included here for the sake of completeness, albeit they are not reflected in our subsequent analysis.

Doersch & Patterson (1977) present a zero-one integer programming formulation of the project scheduling problem which maximizes the NPV of ncf's in a project when progress payments and cash outflows are made upon the completion of certain activities, under the restriction

that capital is rationed. The objective function considers both the cash flows encountered in the performance of the activities and any penalties for late completion of the project. All cash flows occurring within an activity are compounded to the end of the activity allowing a value at completion to be assigned to each activity k, given by

$$v_k = \sum_{j=1}^{d_k} a_{kj} \exp(\alpha(d_k - j)) - c_k \exp(\alpha(d_k)) + c_k \quad [3]$$

where d_k denotes the fixed activity duration, a_{kj} represents the cash flows for activity k in period j ($j=0,1,\ldots,d_k$; $a_{k0} = 0$), e^{α} denotes the compounding rate ($=1/\beta$), and c_k represents the capital investment required by activity k. The objective function has two groups of terms. The first group of coefficients includes the products of the quantity v_k defined by Eq. [3], and a factor required to discount these (compounded) cash flows to the beginning of the project from the finish time implied by the respective zero-one variable, viz.,
$\beta^t = \exp(-\alpha t)$, where t is the time of completion of the activity. The remaining coefficients in the objective function are the delay penalties imposed upon feasible completion times discounted to the beginning of the project by the factor β^t.

The constraints are of three types: (i) activity completion constraints, forcing each activity to finish during the specified project duration, (ii) traditional precedence constraints, and (iii) capital rationing constraints. The capital rationing constraints (one for each time period from the beginning of the project to its latest possible completion time) indicate that: (1) the amount invested in all the activities which can be active during a time period may not exceed the capital available in that time period, which is dependent upon the activities previously scheduled, and (2) sufficient capital must be available to pay any penalties imposed if the project is completed in the period under consideration. Doersch & Patterson report the successful solution of problems consisting of 15 to 20 activities per project using the integer programming code of Geoffrion & Nelson (1968), although projects consisting of more than

30 activites frequently cannot be solved in a reasonable amount of time. Detailed computational results, however, are not provided.

Russell (1986) considers the NPV maximization problem as described by Eqs. [1]-[2], subject to additional resource constraints which ensure that the resource requirements per time period do not exceed the resource availabilities. His concern was not with the optimal scheduling of activities to maximize the project's NPV, which is the problem of concern to us, but rather with the analysis of the consequences of implementing various heuristic rules for activity scheduling subject to resource constraints on the project's NPV. He conducted an experiment in which six heuristic scheduling rules were tested on 80 different problems. One heuristic is the random rule, which is used as a benchmark (selecting the best out of 50 randomly generated solutions). Two heuristics (the minimum slack rule and the minimum latest finishing time rule) were retained mainly because of their success in minimizing project duration in solving the well-known resource-constrained project scheduling problem. The remaining three heuristics are based on the optimal results for the corresponding unconstrained cash flow problem (that is, the problem described by Eqs. [1]-[2] above). As to be expected, no single heuristic performed best on all problems. On small problems, it made little difference which heuristic was used, with the random rule as the best performer (!). The well-known minimal slack rule performed best on the large-scale problems when the resource constraints were not tight. For large scale problems with tight resource constraints, the minimum slack rule was outperformed by one of the three heuristics based on unconstrained cash flow analysis information.

Smith-Daniels & Aquilano (1987) considered the resource-constrained NPV maximization problem assuming that cash outflows occur at the beginning of each activity and a single payment (cash inflow) is received at the completion of the project. Using an extensive set of problems from the literature, they reach the almost self-evident conclusion that a heuristically determined right-shifted schedule (derived from an early-start schedule by right-shifting the activities subject to resource constraints) yields a higher NPV and lower average duration than schedules derived with heuristics that

schedule each activity as early as possible. In addition, while the late-start schedule, on average, was significantly longer than the optimum-duration resource-constrained schedule, no significant difference occurred in the average NPVs of the two scheduling methods.

Smith-Daniels & Smith-Daniels (1987) presented a zero-one programming formulation allowing materials cost and constraints to be added to the basic model of Doersch & Patterson (1977) discussed above. Illustrating the application of the formulation on a small problem example, they do not represent a formalized procedure.

We round up our survey of the NPV maximization problem in project networks, and related contributions, by reviewing some recent research on monetary objective functions in project networks which is tangentially related to the basic problem of concern to us in this paper.

Tavares (1986a) describes the use of a computer simulation program MACAO for estimating the cumulative receipts and disbursements, durations and amounts of consumed nonrenewable resources (such as capital), as well as the probabilities of occurrence of important project events not later than some due dates, from a number of network realizations. In a second paper, Tavares (1986b) discusses the use of a multicriteria scheduling model for estimating the total project duration, the net present value (or total present cost), and the risk of not fulfilling the established schedule, for a large railway renewal program in northern Portugal.

In two more recent contributions, the same author addresses resource-constrained NPV maximization problems which are somewhat different in scope from the ones discussed above. In the first contribution (Tavares 1987a), a dynamic programming (DP) approach is presented for determining the start times and expenditure profiles for a set of interconnected projects (called a program). The objective function to be maximized is a discounted sum of the costs of the project expenditures including a term to penalize the expenditure variation on time. The DP model is realistically applicable only to activities

constrained by strict precedence relations, such as those along the critical path. Except for the railway renewal program already mentioned above, no computational results are given. In the second contribution, Tavares (1987b) presents a new formulation of the resource-constrained project scheduling problem considering the project as a series of stages. According to this approach, the project can be studied as a sequence of decisions on resource allocations in order to maximize the project's NPV. No computational results are provided.

As stated above, our critique and subsequent development shall concentrate on the papers by Russell (1970) and Grinold (1972), and to a lesser degree on the paper by Doersch & Patterson (1977), since it is the solution methodology of the former two papers that are most relevant to our discussion.

3. CRITIQUE AND PROPOSED APPROACH

We are puzzled by the assumption (1) (see section 1) that underlies the work of all researchers in the issue of project NPV, maximum or otherwise, namely, that the cash flow at KE(i), a_i, is independent of the time of realization of KE(i). It seems to us that such an assumption is contrived, and runs counter to common practice. We submit that a more realistic assumption would be that a_i is dependent on the time of realization of KE(i), and should be written as $a_i(T_i)$ to highlight such dependence. If, as it is common in practice, a penalty exists for late delivery of projects or portions thereof, then $a_i(T_i)$ is non-increasing in T_i. The most elementary representation of such a functional relationship would be to assume it linear, but it may possess any other form. In which case the methodologies proposed by the literature are of little help in defining the activities' schedule.

Accepting the two basic assumptions underlying the above referenced developments, we assert that the approaches suggested by Russell (1970) and Grinold (1972) suffer from two serious malaises: (1) they

may fail to achieve a meaningful solution, and (2) they hide the essential simplicity of the problem posed. The following paragraphs elaborate on these two assertions.

That the proposed approaches may yield incorrect, or inconclusive results follows from the fact that, as illustrated below, there are instances in which: (i) the optimal schedule is to delay the project indefinitely, and (ii) any schedule is optimal. Under these conditions, the proposed LP's and the iterative procedures based on them suggested by both authors shall either fail to identify the result unrealistic as it may be, or cycle forever without yielding a definitive answer.

As to the essential simplicity of the problem, we submit that it is rather transparent and is immediately evident from the criterion function, Eq. [1] or Eq.[1']. Indeed, the separability of the objective function in the times of realization $\{T_i\}$ of the various KE's, combined with the fact that the discount factor β ($\equiv e^{-\alpha}$) is < 1, lead one to conclude that the optimal value of T_i is determined by the sign of its coefficient a_i: if positive then T_i should be as small as possible (thus making its term β^{T_i} as large as possible), and if negative then T_i should be as large as possible (thus making its term β^{T_i} as small as possible). How small or large T_i can be is determined, of course, by the precedence constraints Eq.[2]. Viewed from this perspective, the scheduling problem reduces to the problem of either advancing some node realizations or retarding them while respecting the precedence relations. This optic immediately suggests a procedure that is both elementary (in the sense of not appealing to advanced concepts of mathematical formalism such as linear programming and complementary slackness theory) as well as easily implementable on personal computer or, on small problems, by any practitioner armed with pencil and paper.

Before giving the formal definition of our procedure, we solve the two examples provided by Russell and Grinold in their respective papers. The intuitive arguments presented in these solutions will also render the formal enunciation of our procedure more understandable.

Example 1

The small example of Russell (1970), is repeated in Figure 1.

****
Insert Figure 1
****

Since $a_2 < 0$ then T_2 should be as large as possible. And since a_3 and a_4 are both > 0, the realization times T_3 and T_4 should be as small as possible. Assuming the project starts at time $T_1 = 0$, it is immediately evident that the precedence constraints of Eq. [2] result in the tree shown in heavy lines in the figure. Since T_3 and T_4 should be early while T_2 should be late, it immediately follows that respect for the precedence relations results in $T_3 = T_2+4$, and $T_4 = T_2+8$. The criterion function may then be re-written as

$$-5000\ \beta^{T_2} + 3000\ \beta^{T_2+4} + 3000\ \beta^{T_2+8}$$

which may be simplified to

$$\beta^{T_2}\ (-5000 + 3000\ \beta^4 + 3000\ \beta^8)$$

Denote the multiplier of β^{T_2} by V; i.e., let

$$V = -5000 + 3000\ \beta^4 + 3000\ \beta^8 = 651.717$$

where $\beta = e^{-.01} \approx .99$. Since V is positive the value of the objective function is maximized if we put T_2 as small as possible within the defined tree, i.e., $T^*_2 = 4$. Whence $T^*_3 = 8$ and $T^*_4 = 12$. This is the result achieved by Russell via a more elaborate argument.

Because of the simplicity of this example we utilize it to illustrate our contentions above concerning optimal realization times that either (i) escape to infinity, or (ii) are indeterminate.

Suppose, in the above example of Russell, that $a_3 = 0$ and $a_4 = 5350$. Then an argument similar to above leads to the criterion function:

$$-5000\ \beta^{T_2} + 5350\ \beta^{T_2+8} = \beta^{T_2}\ V$$

where

$$V = -5000 + 5350\,\beta^{8} = -61.33$$

Now that the coefficient of β^{T_2} is < 0, the value of the objective function is maximized by letting $T_2 \dashrightarrow \infty$, a highly unrealistic conclusion. More to the point, what was almost immediately obvious may have consumed considerable effort to fathom via the procedures of Russell and Grinold.

Finally, suppose again that in the above example $a_3 = 0$ but $a_4 = 5000/\beta^{8} = 5416.435$. Then, evidently, the criterion function would reduce to:

$$-5000\,\beta^{T_2} + (5000/\beta^{8})\beta^{T_2+8} = \beta^{T_2}\,V$$

where

$$V = -5000 + (5000/\beta^{8})\beta^{8} = 0$$

Since the coefficient of β^{T_2} is zero any value of T_2 is optimal. Grinold's procedure would have cycled forever (because in step 2 of his procedure the values of: θ, δ, PV, T_1 and w_{ij} will remain unchanged, such that in step 3 the value of θ will be equal to 0, leaving $w_{N1} > 0$, whence the stopping rule in step 1 of his algorithm -
$w_{N1} = 0$ and the current schedule is optimal for all durations beyond the current value of δ - will not apply).

Example 2

Grinold's example is shown in Figure 2, whose analysis proceeds as follows.

****
Insert Figure 2a and 2b
****

Since a_2, a_3, and a_6 are all < 0 then it is advantageous (relative to the maximization of the NPV criterion) to have these three nodes realized as late as possible. On the other hand, since a_4, a_5, and a_7

are all > 0, we desire their realization time to be as early as possible. Slight reflection results in a forest composed of the two trees shown in heavy lines in Figure 2a, denoted by t_1 and t_2:

t_1: nodes {2,3,4,5}, arcs {(2,4),(2,5),(3,5)}
times : T_2, $T_4 = T_2 + 1$, $T_5 = T_2+5$, $T_3 = T_5-4 = T_2+1$
t_2: nodes {6,7}, arc {(6,7)}, times : T_6, $T_7 = T_6+5$

Let T_6 be the unknown variable in the tree t_2. The NPV of this tree is given by

$$-200\ \beta^{T_6} + 300\ \beta^{T_6+5} = \beta^{T_6}\ V$$

where

$$V = -200 + 300\ \beta^5 = 85.37$$

which indicates that, for maximal value, T_6 should be as small as possible. This fixes the location of the tree t_2 relative to the tree t_1 through arc (3,6) since, relative to node 6 we have

$\max \{T_3+8, T_4+3\} = \max \{T_2+1+8, T_2+1+3\} = T_2+9$ through arc (3,6), and relative to node 7 we have

$\max \{T_6+5, T_5+8\} = \max \{T_2+9+5, T_2+5+8\} = T_2+14$ through arc (6,7).

Tree t_2 is now connected to tree t_1 via arc (3,6), to result in a single tree t over the graph of the project network as shown in Figure 2b. The only unknown variable in t is T_2, with objective function given by:

$$\beta^{T_2}\ (-200 - 200\ \beta + 100\ \beta + 400\ \beta^5 - 200\ \beta^9 + 300\ \beta^{14})$$

$$= \beta^{T_2}\ V$$

where

$$V = -200 - 200\ \beta + 100\ \beta + 400\ \beta^5 - 200\ \beta^9 + 300\ \beta^{14}$$
$$= 159.51$$

in which the terms correspond to nodes 2,3,...,7, respectively. Consequently, it is advantageous to make T_2 as small as possible.

Tree t now connects to node 1 through arc (1,2), yielding the times of realization:

$T^*_1 = 0$, $T^*_2 = 3$, $T^*_3 = 4$, $T^*_4 = 4$, $T^*_5 = 8$, $T^*_6 = 12$, and $T^*_7 = 17$. This is precisely the result obtained by Grinold via mathematical programming and network flow arguments.

We now formalize the intuitive arguments presented above in a formalized procedure.

4. AN OPTIMAL PROCEDURE FOR MAXIMIZING THE NPV IN ACTIVITY NETWORKS

To simplify notation we use generic designation of nodes and sets. At any step of the procedure we shall be concerned with a particular tree over a subset of nodes. We shall distinguish between the tree defined on these nodes, which is generically denoted by t_q, q = 1,2,..., and the set of nodes, which is generically denoted by $J(t_q)$ or simply **Jq**. Each tree has a seed (or root) node, generically designated by $s(t_q)$, or simply s(q). The time of realization of any seed node is either indeterminate, in which case s(q) must be a KE with negative ncf which is not immediately preceded by any KE with negative ncf, or the time of realization of the seed s(q) is determinate, in which case the seed node must be node 1. It immediately follows that all KE's with determinate times of realization belong to the same tree with root at node 1. We identify this tree as tree t_D. Other trees, if they exist, must originate at KE's with negative ncf's. The time of realization of any KE(i) in a tree is represented as the sum of the (unknown) time of realization of the seed node s(q) and a constant equal to the length of the longest chain from s(q) to that node, denoted by d_{si}. We use the word chain advisedly, since the sequence of arcs in **t** from s(q) to KE(i) may contain forward as well as reverse arcs, as shall be seen presently. Thus all the nodes of a tree **t** will belong to the same class: either all have determinate times of realization (in the tree t_D), or all

have indeterminate times of realization that are represented as $\min(T_s) + d_{si}$.

We borrow the term critical path (cp) from the vernacular of classical CPM to designate the length of the longest path from some set J_q to KE(i) ignoring all cash flow considerations.

Although the two simple examples solved above did not demonstrate it, there is need for the temporary allocation of KE's to sets, and the temporary determination of their times of realization, both of which may later be modified. This need arises when a KE(i) has positive ncf and belongs to a tree t with positive ncf and is linked to another tree t' with negative cash flow, with the cp from t longer than the earliest realization time from t'. In this case a displacement interval R has to be computed during which the corresponding node realization times remain valid.

4.1 Procedure NPV

0. Define two classes of sets: the sets $\{J_q\}$, q = 1,2,... for nodes with indeterminate times of realization, and the set J_D for nodes with determinate times of realization. Start with $J_q = \phi$ for all q, including D.

THE FOREST GENERATION STEP

1. Consider the start node i = 1.
   If $a_1 >= 0$, add node i=1 to the determinate tree:
   $J_D = \{1\}$
   $V_D = a_1$
   $T_1 = 0$

Else, let q = 1 and create a new indeterminate tree t_1 with node 1 as the starting node:

$J_1 = \{1\}$

$V_1 = a_1$

$T_1 \equiv T_{s(1)} = 0$, indeterminate

2. Do for i = 2,..,n

If $a_i < 0$, let q = q+1 and create a new indeterminate tree t_q with root node s(q) = i:

$Jq = \{i\}$

$V_q = a_i$

$T_i \equiv T_{s(q)}$, with $T_{s(q)} = \min(T_{s(q)})$, that is, the earliest realization time of KE(i) under standard CPM calculations.

Else, determine the longest chain into node i from each of its immediate predecessor nodes (If several longest chains exist, choose one at random). Let d_{ji} denote the length of the longest chain connecting from node j belonging to tree t_q (determinate or indeterminate). Add node i to tree t_q:

$T_i = T_j + d_{ji}$

$J_q = \{J_q\} \cup \{i\}$

$V_q = V_q + a_i \beta^{T_i - T_{s(q)}}$

If Vq <= 0, go to next node i=i+1

Else, check each node m of tree t_q for an immediate predecessor node $j \in J_q$. belonging to a tree t_q. with negative V_q..

If no such predecessor node can be found, go to next node i=i+1

Else, compute for each such predecessor node $j \in J_q$. (belonging to tree t_q.) the displacement interval $R_q. = T_m - T_j - d_{jm}$, and determine the corresponding tree t_k for which the displacement interval, R_k, is the smallest.

If this minimal displacement interval R_k affects the earliest CPM realization times of nodes beyond the realization time of the current node i, R_k may not be optimal. Check R_k for each of its time increments:

The node realization times T_j of the nodes $j \in J_k$ belonging to tree t_k are updated for each time increment of the displacement interval R_k and the earliest realization times of all the network nodes are re-computed accordingly, yielding a NPV of the network for each time increment of R_k. Select the R_k value which yields the maximum NPV.

If the minimal displacement interval R_k is optimal, create a temporary tree t_t consisting of all the nodes belonging to tree t_q., node m itself, and all the successors of node m in tree t_q. Compute, V_t, the NPV of this temporary tree.

If $V_t >= 0$, establish the temporary link and update $Jq = J_q \cup J_q.$; otherwise, $J_q. = J_t - \{$node m and all its successors$\}$.

If $V_q > 0$, repeat.

THE TREE COALESCING STEP

Find the tree t_Q containing the terminal node n. Denote the number of nodes in t_Q by count(t_Q).

Do While count(t_Q) < n

$$\text{If } V_Q = \sum_{i \in J_Q} a_i \beta^{(Ti - T_{s(Q)})} > 0$$

determine the smallest shift backwards in time (towards the zero time) of tree t_Q until it links with another tree t_q. Coalesce the trees: $J_Q = J_Q \cup J_q$.

$$\text{If } V_Q = \sum_{i \in J_Q} a_i \beta^{(Ti - T_{s(Q)})} < 0$$

determine the largest shift forwards in time (away from the zero time) until it links with another tree t_q. Coalesce the trees: $J_Q = J_Q \cup J_q$. If no such tree exists, then the problem is infeasible.

Else, try to coalesce tree t_Q with the the closest tree t_q that occurs earlier than it (according to the convention of earliest completion time).

4.2 Illustrative example

Consider the project represented by the network of Figure 3, with the following ncf's: $a_1 = 0$, $a_2 = -100$, $a_3 = -180$, $a_4 = +40$, $a_5 = +40$, $a_6 = -50$, $a_7 = +100$, $a_8 = -40$, $a_9 = +200$, $a_{10} = +90$, $a_{11} = +500$.

****
Insert Figure 3
****

Prima facie, the project is profitable since the sum of positive ncf's exceeds the sum of negative ncf's by 600. Unfortunately, such simplistic analysis does not take into account the time value of money.

The steps of analysis following the Procedure NPV are detailed below.

1. Node 1 with $a_1 = 0$ is added to the determinate tree:
create set $J_D = \{1\}$, $s(D) = 1$, $V_D = 0$, $T_1 = 0$.

2. Node 2 has $a_2 < 0$. Create a new tree t_1: $J_1 = \{2\}$,
$s(1) = 2$, $V_1 = -100$. The time T_2 is indeterminate with $\min(T_2) = 20$.

3. $a_3 < 0$: a new tree t_2 is created. Create the new set $J_2 = \{3\}$;
$s(2) = 3$; $V_2 = -180$. The time T_3 is indeterminate, with $\min(T_3) = 50$.

4. Node 4 has $a_4 > 0$. Node 4 has two immediate predecessors: node $3(\in J_2)$ and node $1(\in J_D)$.
Node 3 ==> $\min(T_3)+45 = 95$
Node 1 ==> $d_{1,4} = 100$
Node 4 is added to the determinate tree t_D as the successor of node 1: $J_D = \{3,4\}$, $V_D = 14.64$, $T_4 = 100$.

Node 4 is the immediate successor of node $3(\in J_2)$ and $V_2<0$. The displacement interval is given by $R_2=100-95=5$, indicating that tree t_2 can be moved away from zero time by at most 5 time units. Re-computing $\min(T_3)= 55$ affects the earliest realization time of node 6: $\min(T_6) = 85$. The NPV of the network is checked for each time increment of the displacement interval. $R_2 = 5$ is kept as the best value. Create a tempory tree consisting of all nodes belonging to tree t_2 (i.e., node 3), node 4 itself, and all its successors (none). Since this tree has a negative NPV: drop node 4 from t_D. Set
$J_D = J_D-\{4\} = \{1\}$,
$s(D) = 1$, $V_D = 0$, $T_1 = 0$; $J_2 = \{3,4\}$, $s(2) = 3$,
$V_2 = -154.55$, $\min(T_3) = 55$, $\min(T_4) = \min(T_3)+45 = 100$. Keep $\min(T_6)=85$. Tree t_1 remains unchanged:
$J_1 = \{2\}$, $s(1) = 2$, $V_1 = -100$, $\min(T_2) = 20$. The corresponding forest is represented in Figure 4.

****
Insert Figure 4
****

5. Node 5 has $a_5 > 0$. Node 5 has two predecessors: node 3 and node 4, both in J_2. Add node 5 to J_2 as the immediate successor of node 4: $J_2 = \{3,4,5\}$, $s(2) = 3$,
$V_2 = -139.16$, $\min(T_5) = \min(T_3)+95 = 150$.

6. $a_6 < 0$: create a new tree t_3. Set $J_3 = \{6\}$, $s(3) = 6$,
$V_3 = -50$, $\min(T_6) = 85$.

7. $a_7 > 0$. Node 7 has three immediate predecessors:
nodes 3 and 4 are both in J_2 ==> $\min(T3)+d_{37} = 140$;
node 6 is in J_3 ==> $\min(T_6) + d_{67} = 120$.
Consequently, node 7 is added to J_2 : $J_2 = \{3,4,5,7\}$,
$V_2 = -96.6$, $s(2) = 3$, $\min(T_7) = 140$.

8. Since $a_8 < 0$, a new tree t_4 is created: $J_4 = \{8\}$,
$s(4) = 8$, $V_8 = -40$, T_8 is indeterminate with $\min(T_8) = d_{18} = 170$.

9. $a_9 > 0$. Node 9 has three immediate predecessors:
nodes 5 and 7 in set J_2 ==> $\min(T_3)+d_{39} = 195$;
node 8 in set J_4 ==> $\min(T_8)+d_{89} = 170+15 = 185$.
Consequently, node 9 is added to tree t_2:
$J_2 = \{3,4,5,7,9\}$, $s(2) = 3$, $V_2 = -47.62$, $\min(T_9) = 195$.

10. $a_{10} > 0$. Node 10 has two immediate predecessors, 5 and 9, both of which are in J_2. Add node 10 to t_2 as the immediate successor of node 9: $J_2 = \{3,4,5,7,9,10\}$,
$s(2) = 3$, $V_2 = -28.67$, $\min(T_{10}) = 210$. The corresponding forest is given in Figure 5.

****
Insert Figure 5
****

11. $a_{11} > 0$. Node 11 has three immediate predecessors:
node 8 in J_4 ==> $\min(T_8)+d_{8,11} = 170+60 = 230$;
nodes 9 and 10 in J_2 ==> $\min(T_3)+d_{3,11} = 240$ via node 10.

Consequently, add node 11 to t_2 as the successor of node 10: J_2 = {3,4,5,7,9,10,11}, $\min(T_{11}) = 240$, $V_2 = 49.22$. Node 9 in tree t_2 is the immediate successor of node 8 in tree t_4 which has a negative V_4. The displacement in node 8 is obviously limited by the time of realization of node 9, implying that node 8 can be delayed only 10 time units to realize node 9 at time $\min(T_9) = 195$. R_4 is computed as $R_4 = 195-(170+15) = 10$, $\min(T_8) = 180$. Create a temporary tree consisting of all the nodes in tree t_4 , node 9 itself, and all the successors of node 9 (nodes 10 and 11). Since the NPV of this tree is positive, the link is established:J_2 = {3,4,5,7,8,9,10,11}, s(2) = 3, $V_2 = 37.83$. The resulting forest is given in Figure 6.

****
Insert Figure 6
****

Since V_2 is still positive, we check for further predecessors: node 7 belonging to tree t_2 is an immediate successor of node 6, belonging to tree t_3 with $V_3 < 0$. It is clear that the delay in node 6 is limited by the time of realization of node 7, implying that node 6 can be delayed only 20 time units to realize T_7 at $\min(T_7)$=140. Compute the displacement interval as R_3 =140-(85+35)=20. This is the optimal displacement: keep $\min(T_6)$=105. Establish the link between tree t_2 and tree t_3:
J_2 = {3,4,5,6,7,8,9,10,11}, s(2) = 3, $V_2 = 7.58$. The resulting forest is given in Figure 7.

****
Insert Figure 7
****

Since $V_2 > 0$, we continue checking for predecessors. Node 6 in tree t_2 is the immediate successor of node 2 in tree t_1 which has $V_1 < 0$. It is clear that the delay in node 2 is limited by the time of realization of node 6, implying that node 2 can be delayed only 60 time units to realize T_6 at $\min(T_6) = 105$. The displacement $R_1 = 60$ is optimal: keep $\min(T_2) = 80$, compute the net present value of the temporary tree consisting of node 2 in t_1, node 6 and the immediate successors of node 6 in tree t_2. Since this NPV is negative, drop node 6 and 7 from t_2 and add them to tree t_1 : J_1 = {2,6,7}, s(1) = 2, $V_1 = -84.18$;

J_2 = {3,4,5,8,9,10,11}, s(2) = 3, V_2 = -4.73. The forest is given in Figure 8.

****
Insert Figure 8
****

We now proceed with the coalescing step. The tree containing the final node is tree t_2 with $V_2 < 0$. When it is moved forwards in time (away from zero time), it links with tree t_1 because of the arc between nodes 4 and 7. Coalesce both trees: J_1 = {2,3,4,5,6,7,8,9, 10,11},
s(1) = 2, V_1 = -70.20. Tree t_1 can now be moved away from zero time without coalescing with another tree. If event 1 (the start of the project) in tree t_1 is realized at time 0, then the rest of the events 2 through 11 shall break loose and be realized as late as possible; i.e., delayed without bound. There is no feasible optimal solution.

5. COMPUTATIONAL RESULTS

Procedure NPV has been programmed in Microsoft C, Version 5.10 for the IBM PS/2 Model 70 (or compatibles) running under DOS. The program has been tested on a series of test problems generated by the activity network generator described in Herroelen et al. (1989). A total of 250 random activity-on-the-arc networks have been generated as follows.

The number of nodes (events) ranges from 5 to 20. For a given number of nodes n, five values for the number of activities were drawn from the range n-1 (the single chain) to n(n-1)/2 (the completely connected network). As such the smallest network considered had 5 nodes and 4 arcs, while the largest network included had 20 nodes and 190 arcs. Activity durations were randomly generated by drawing from an exponential distribution with a mean equal to 20 time units. The a_i values were randomly generated from a normal distribution with a mean of 60 and a standard deviation of 20, and based on a probability of 1/8 for a cash flow of zero, 3/8 for a negative and 4/8 for a positive cash flow.

For each of the 50 node-activity combinations, 5 networks were generated, yielding a total of 250 networks. Each network was executed 100 times. CPU time (excluding input and output) was measured in thousandths of a second and averaged over the 100 network executions. The computational results are given in Table I.

****
Insert Table I
****

Each problem was also solved to optimality by implementing the linear programming procedure, reviewed earlier in this paper and described by Russell (1970), in Super LINDO (Schrage 1989). The Super LINDO version at our disposal was equipped with a timer routine capable of measuring time in hundredths of a second. This time includes the CPU time involved in pure processing and program output (not to be avoided since program output is part of the GO command in LINDO), but does not include the time needed to set up the LP formulation. The computational results are also given in Table I.

Procedure NPV optimally solved all problems in very reasonable CPU-times. The largest CPU time reported was for a 20 nodes - 63 activities problem where it took some 21.181 seconds (averaged over 100 network executions) to reach the optimal solution. It took LINDO almost 2 1/2 minutes to solve the same problem. On the other hand, a 20 nodes - 146 activities problem required almost 13 minutes using LINDO, while it could be solved in 0.1835 seconds by Procedure NPV. Results seem to indicate that it is the combination of the network structure and the cash flow pattern that may add to the complexity of a problem and not that much the size of the problem in terms of the number of activities and events.

6. CONCLUSIONS

In this paper we critically reviewed the approaches offered for the resolution of the problem of maximizing the net present value of a project through the manipulation of the times of realization of its key events. All the procedures offered in the literature, including

Procedure NPV described in this paper, take the fundamental assumptions that the ncf's a_i are independent of the time of realization of the critical events KE(i), and are known a priori. It would be a valid area for further research to investigate the project NPV problem under the more realistic assumption that the net cash flows a_i are dependent on the time of realization of the critical events, and are related to the outcome of the project bid.

For the problem of maximizing the project NPV in the absence of other than traditional precedence constraints, it was found that previous developments by Russell (1970) and Grinold (1972) may fail to achieve a meaningful solution and essentially hide the simplicity of the problem. In this paper an optimal procedure was developed based on the transparent solution strategy of advancing or retarding node realizations subject to the activity precedence constraints. As such the procedure consists of a forest generation and tree coalescing step. The solution procedure has been programmed for personal computer and validated on an extensive series of randomly generated test problems. Having solved all problems in acceptable CPU times, computational results are promising.

REFERENCES

DANTZIG, G.B. (1963), **Linear Programming and Extensions**, Princeton University Press, New Jersey.

DOERSCH, R.H. and J.H. PATTERSON (1977), Scheduling a Project to Maximize its Present Value: A Zero-One Programming Approach, **Management Science**, 23(8), 882-889.

ELMAGHRABY, S.E. (1977), **Activity Networks: Project Planning and Control by Networks**, John Wiley &Sons, New York.

GEOFFRION, A.M. and A.B. NELSON (1968), User's Instructions for 0-1 Integer Linear Programming Code RIP30C, Memorandum RM-5657-PR, The Rand Corporation.

GRINOLD, R.C. (1972), The Payment Scheduling, **Naval Research Logistics Quarterly**, 19(1), 123-136.

HERROELEN, W.S, E. DEMEULEMEESTER and B. DODIN (1989), The Generation of Strongly-Random Activity Networks, Research Report N° 8901, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium.

RUSSELL, A.H. (1970), Cash Flow in Networks, **Management Science**, 16(5), 357-373.

RUSSELL, R.A. (1986), A Comparison of Heuristics for Scheduling Projects with Cash Flows and Resource Restrictions, **Management Science**, 32(10), 1291-1300.

SCHRAGE, L. (1989), **Linear, Integer and Quadratic Programming with LINDO**, The Scientific Press.

SMITH-DANIELS, D.E. and N.J. AQUILANO (1987), Using a Late-Start Resource-Constrained Project Schedule to Improve Project Net Present Value, **Decision Sciences**, 18, 617-630.

SMITH-DANIELS, D.E. and V.L. SMITH-DANIELS (1987), Maximizing the Net Present Value of a Project Subject to Materials and Capital Constraints, **Journal of Operations Management**, 7(1-2), 33-45.

TAVARES, L.V. (1986a), Stochastic Planning and Control of Program Budgeting - The Model MACAO, **OR Models and Microcomputers**, Coelho & Tavares (eds.), Elsevier Science Publishers B.V., North-Holland.

TAVARES, L.V. (1986b), Multicriteria Scheduling of a Railway Renewal Program, **European Journal of Operational Research**, 25, 395-405.

TAVARES, L.V. (1987a), Optimal Resource Profiles for Program Scheduling, **European Journal of Operational Research**, 29, 83-90.

TAVARES, L.V. (1987b), A Multi-Stage Non-Deterministic Model for Project Scheduling Under Resource Constraints, Research Paper, Center of Urban and Regional Systems of the Technical University of Lisbon.

FIGURE CAPTIONS

Figure 1. Russell's example.

Figure 2. Grinold's example.

Figure 3. Illustrative project network example.

Figure 4. Forest consisting of the determinate tree and two indeterminate trees.

Figure 5. Updated forest.

Figure 6. Forest for the 11 nodes of the example problem.

Figure 7. Forest update.

Figure 8. Forest update.

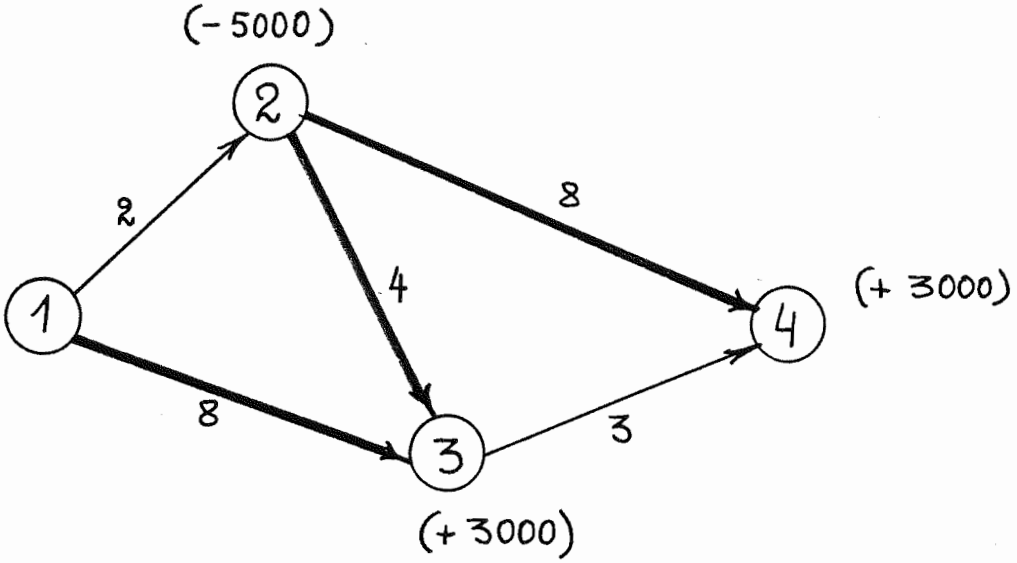

Fig. 1

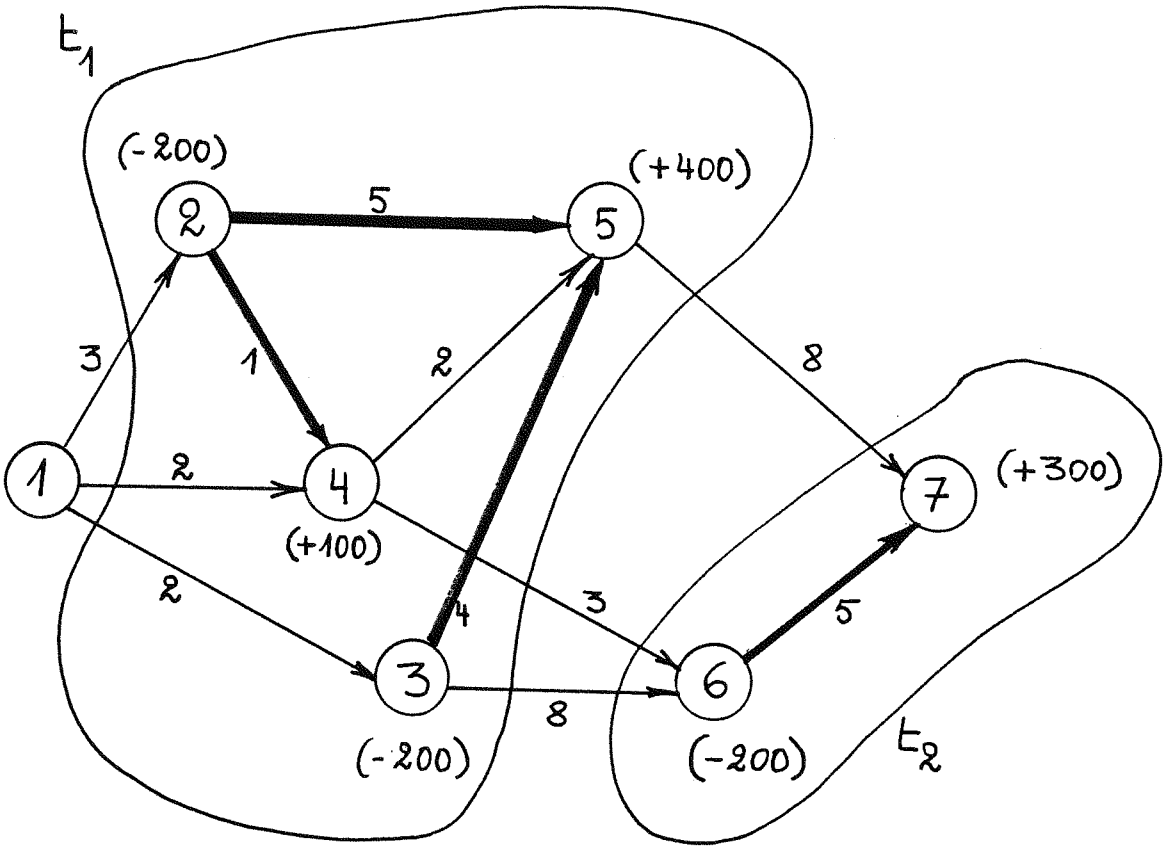

Fig. 2a

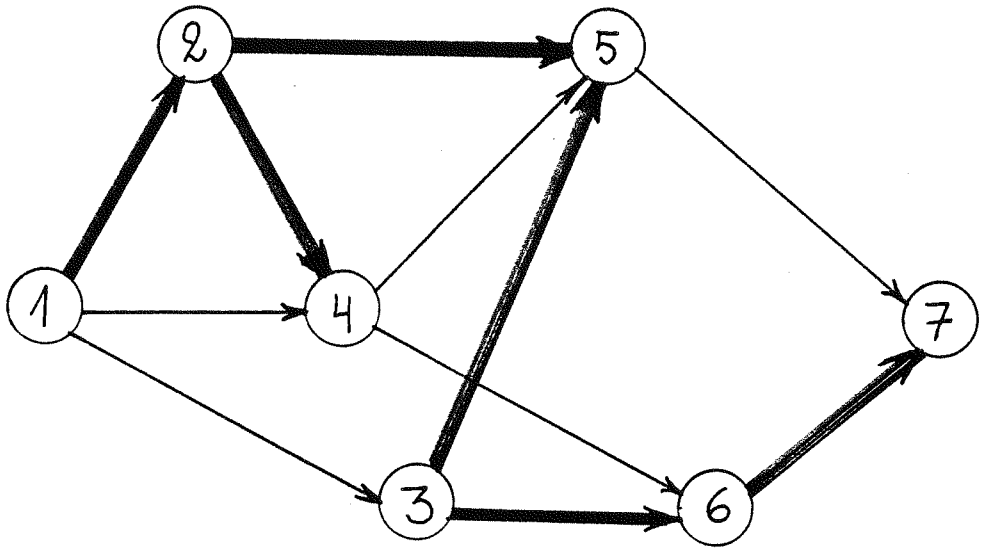

Fig. 2b.

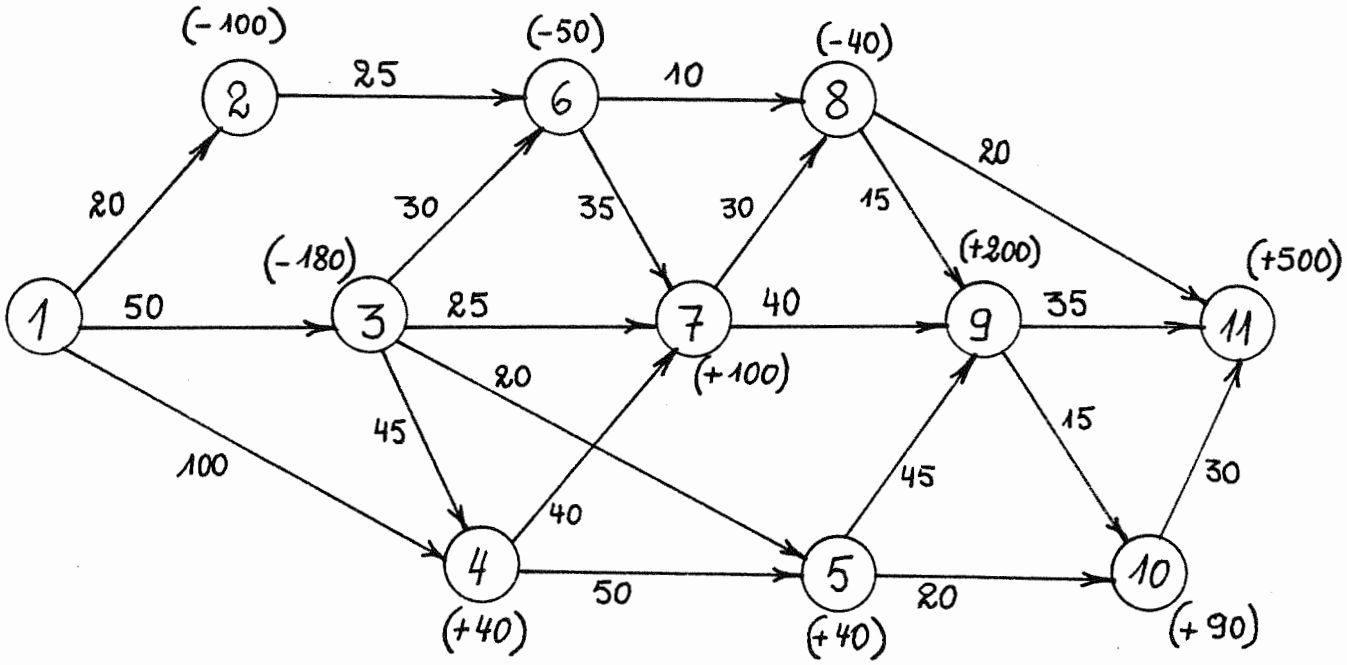
(-100)
(-50)
(-40)
2
6
8
25
10
20
20
30
35
30
15
(-180)
(+200)
(+500)
1
50
3
25
7
40
9
35
11
20
(+100)
45
15
100
45
30
40
4
5
10
50
20
(+40)
(+40)
(+90)

Fig. 3

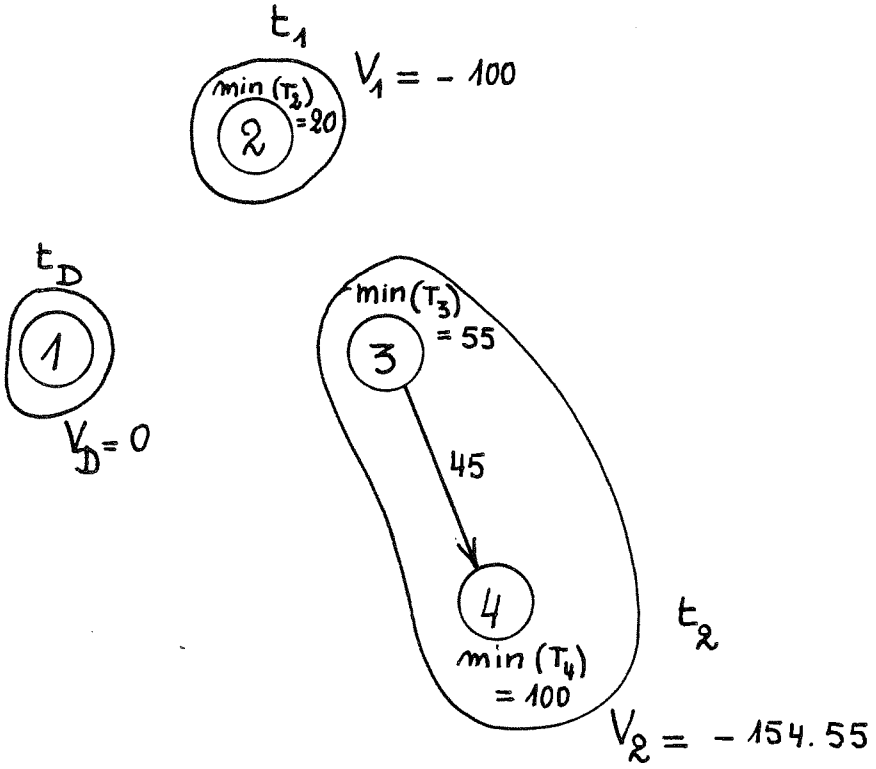

Fig. 4

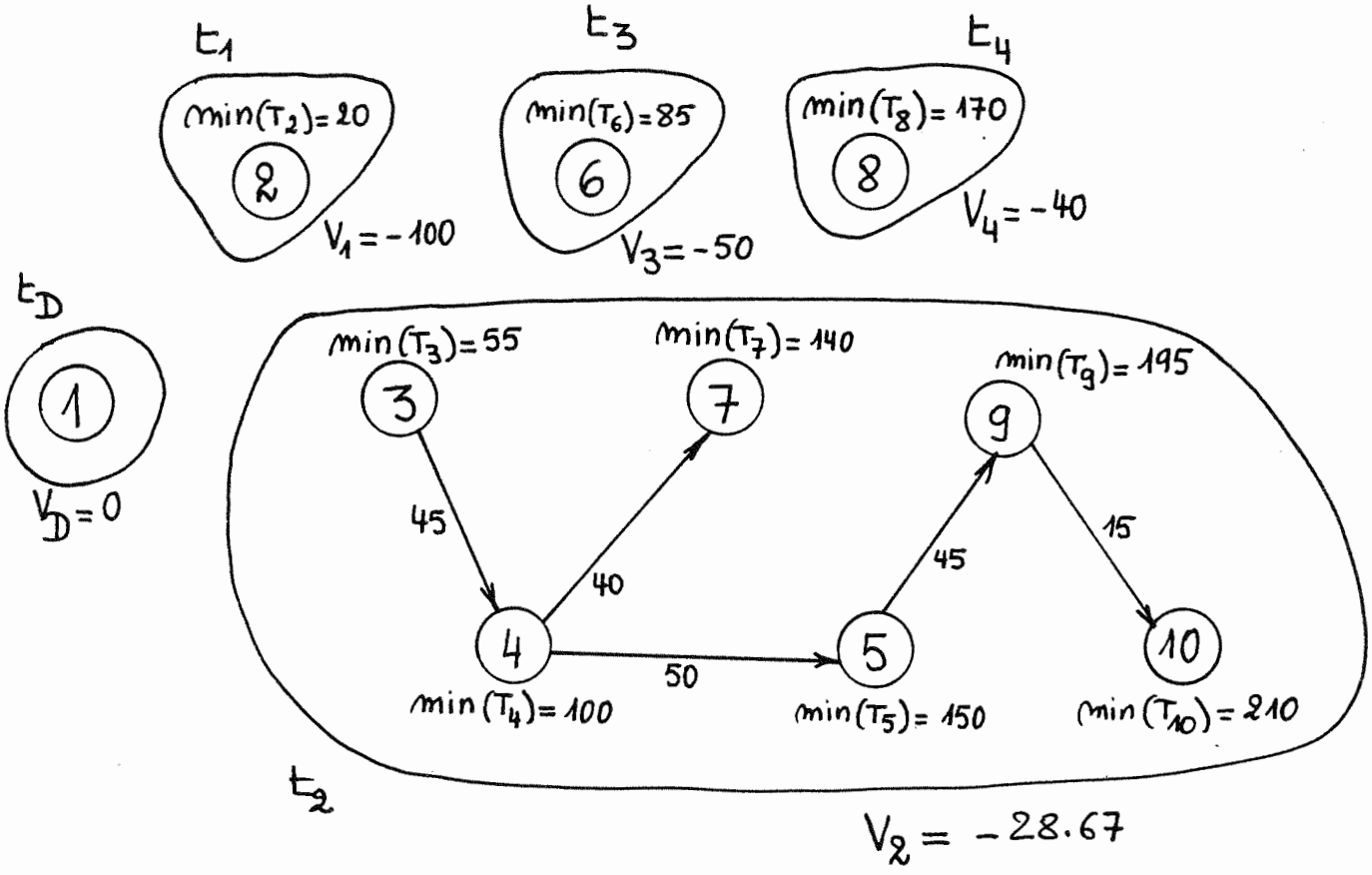
t1
min(T2)=20
2
V1=-100
t3
min(T6)=85
6
V3=-50
t4
min(T8)=170
8
V4=-40
tD
1
VD=0
min(T3)=55
3
min(T7)=140
7
min(T9)=195
9
45
40
45
15
4
50
5
10
min(T4)=100
min(T5)=150
min(T10)=210
t2
V2=-28.67

Fig. 5

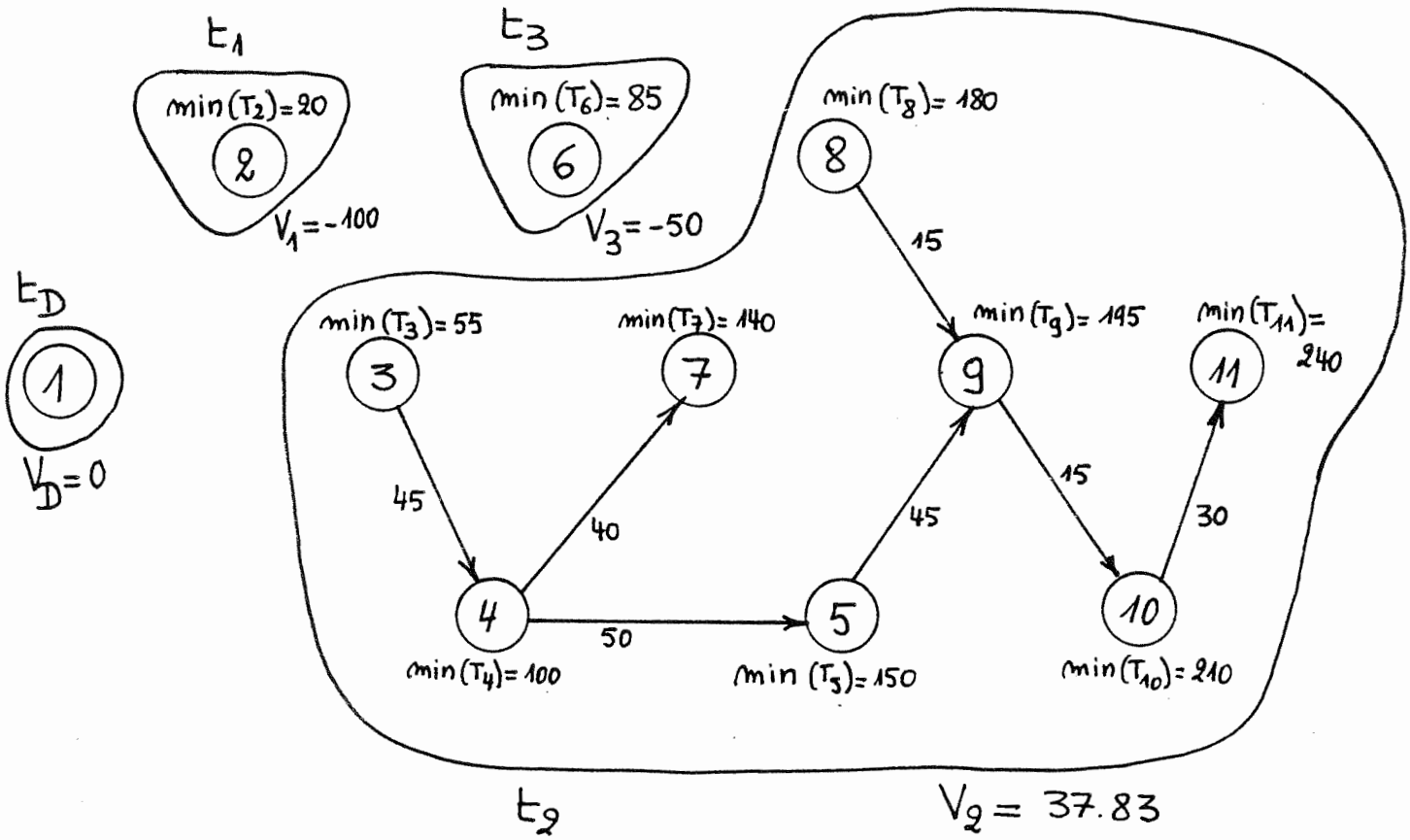
t1
min (T2) = 20
2
V1 = -100
t3
min (T6) = 85
6
V3 = -50
tD
1
VD = 0
min (T8) = 180
8
15
min (T3) = 55
3
min (T7) = 140
7
min (T9) = 195
9
min (T11) = 240
11
45
40
45
15
30
4
50
5
10
min (T4) = 100
min (T5) = 150
min (T10) = 210
t2
V2 = 37.83

Fig. 6

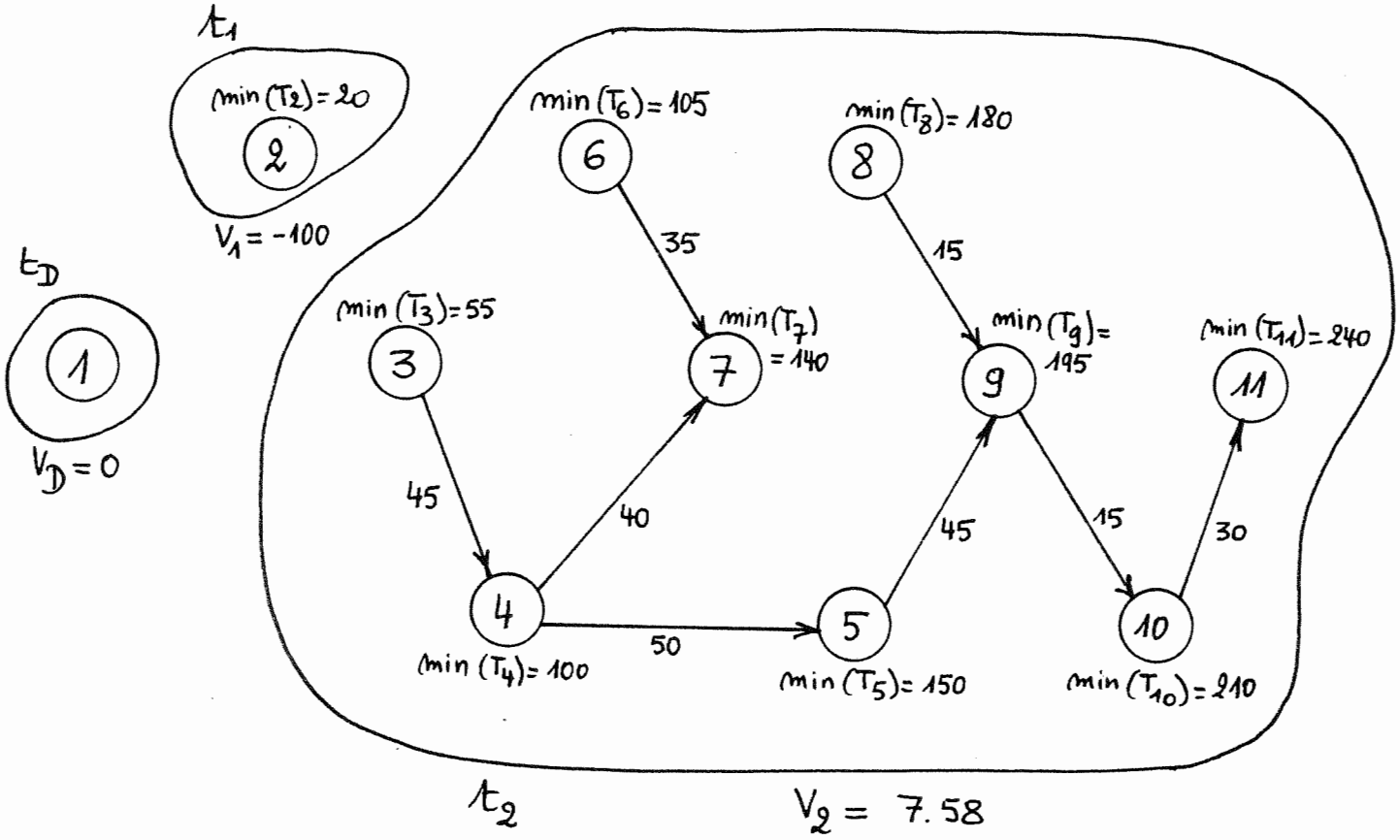
t_1
$\min(T_2) = 20$
2
$V_1 = -100$
t_D
1
$V_D = 0$
$\min(T_6) = 105$
6
35
$\min(T_8) = 180$
8
15
$\min(T_3) = 55$
3
$\min(T_7) = 140$
7
$\min(T_9) = 195$
9
$\min(T_{11}) = 240$
11
45
40
45
15
30
4
50
5
10
$\min(T_4) = 100$
$\min(T_5) = 150$
$\min(T_{10}) = 210$
t_2
$V_2 = 7.58$

Fig. 7

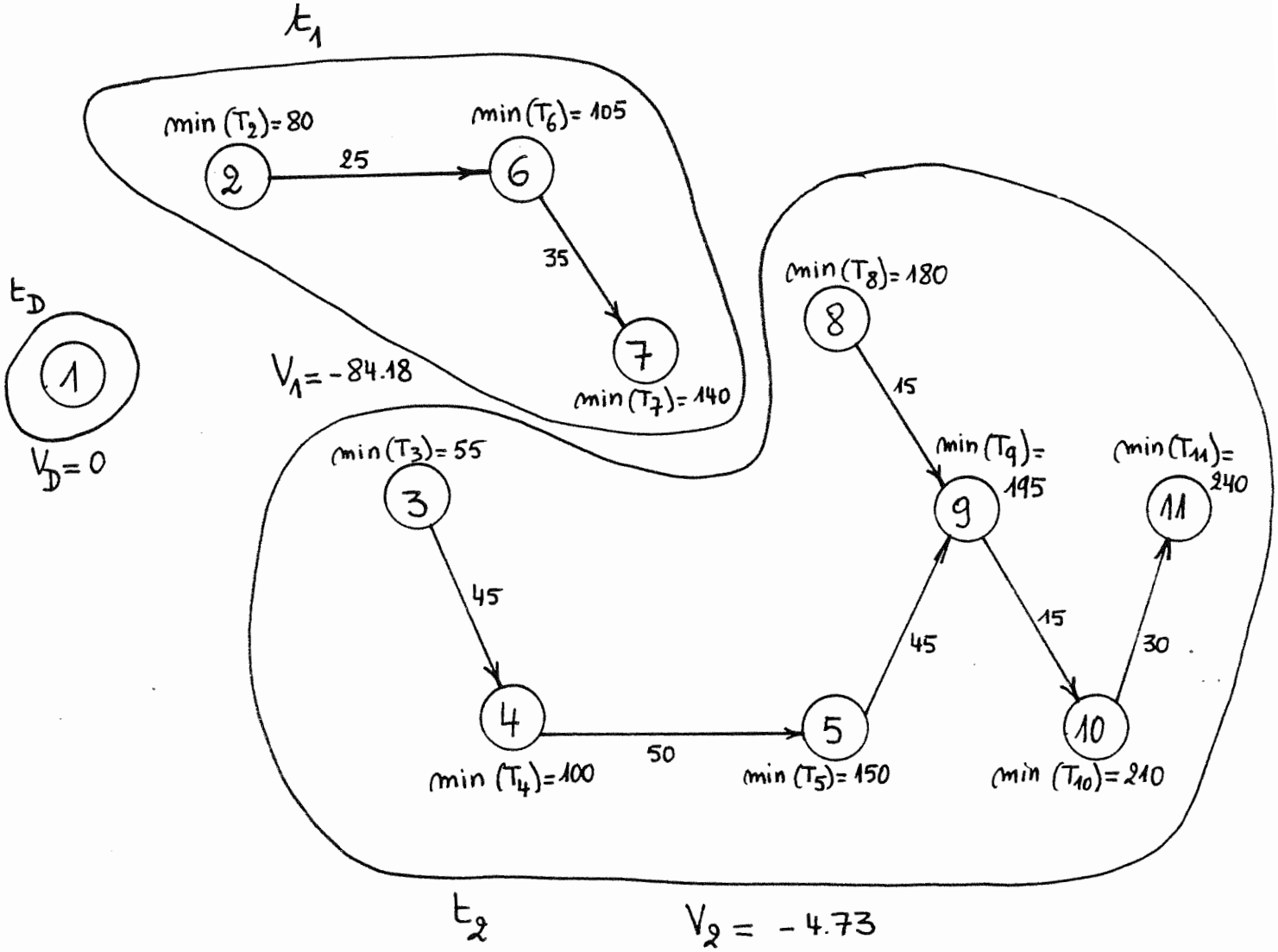
t_1
$\min(T_2) = 80$
$\min(T_6) = 105$
2
25
6
35
7
$\min(T_7) = 140$
$V_1 = -84.18$
t_D
1
$V_D = 0$
$\min(T_8) = 180$
8
15
$\min(T_9) = 195$
9
$\min(T_{11}) = 240$
11
$\min(T_3) = 55$
3
45
4
50
5
45
15
30
10
$\min(T_4) = 100$
$\min(T_5) = 150$
$\min(T_{10}) = 210$
t_2
$V_2 = -4.73$

Fig. 8

TABLE CAPTIONS

Table I. CPU times obtained by Procedure NPV and LINDO.

Number of nodes	Range number of activities	Procedure NPV (average CPU time in secs)	Super LINDO (average CPU in minutes:secs)
5	4 - 10	0.0309	0: 2.356
6	5 - 15	0.0308	0: 3.312
8	7 - 28	0.1535	0: 6.157
10	9 - 45	0.0773	0:11.687
12	11 - 66	0.2786	0:22.907
14	13 - 91	0.4349	0:39.510
16	15 - 120	0.4055	1: 3.368
18	17 - 153	1.3569	1:46.128
19	18 - 171	0.9584	3: 2.460
20	19 - 190	1.8922	3:48.261